CLAIMS:

1. A method of processing graphics data using a buffer comprising:

receiving fragments, a fragment associated with a location in the buffer;

tracking a pending write to the location in the buffer;

shading at least a portion of the fragments to produce shaded fragment data;

waiting to read the location in the buffer until the pending write to the location in the buffer is completed;

shading the fragment using data read from the location in the buffer to produce additional shaded fragment data;

writing the shaded fragment data to at least one location in the buffer; and

writing the additional shaded fragment data to a location in the buffer.

2. The method of claim 1, wherein the tracking comprises:

entering the location associated with the pending write in a conflict detection unit; and

updating the conflict detection unit when the pending write to the location is completed.

3. The method of claim 1, wherein the tracking comprises:

entering a region containing the location associated with the pending write in a conflict detection unit; and

updating the conflict detection unit when the pending write to the location is completed.

4. The method of claim 1, wherein data stored in the location in the buffer is also stored in an entry in a data cache.

5. The method of claim 4, further comprising:

invalidating the entry in the data cache associated with the pending write to the location in the buffer.

6. The method of claim 4, further comprising:

updating the entry in the data cache when the pending write to the location is completed.

7. A method for processing fragments under control of a fragment program in a fragment processing unit, comprising:

determining a write to a location in a buffer is pending prior to reading the location in the buffer;

waiting for the write to complete;

reading the location in the buffer; and

processing a fragment in the fragment processing unit as specified by the fragment program.

8. The method of claim 7, wherein the fragment program performs depth buffering prior to shading.

9. The method of claim 8, wherein the fragment program performs depth peeling.

10. The method of claim 8, further comprising:

processing another fragment as specified by the fragment program while waiting for the write to complete.

11. The method of claim 8, wherein the buffer is one of several buffers stored in graphics memory.

12. A programmable graphics processor for execution of program instructions comprising:

a conflict detection unit configured to selectively store at least a portion of a position associated with a fragment and generate a position conflict status;

a read interface configured to read data associated with the position from a graphics memory and output the data to a fragment processing unit;

the fragment processing unit configured to receive a fragment associated with the position, and the data from the read interface and generate a processed fragment; and

a write interface configured to write the processed fragment to the graphics memory.

13. The programmable graphics processor of claim 12, wherein the portion of a position specifies a region of fragment positions.

14. The programmable graphics processor of claim 12, wherein the read interface is configured to read data responsive to the position conflict status.

15. The programmable graphics processor of claim 12, wherein a position stored in the conflict detection unit includes at least a buffer identifier and a pair of coordinates.

16. The programmable graphics processor of claim 12, wherein the fragment processing unit further includes a data cache configured to store data entries, each data entry associated with a position in a buffer.

17. The programmable graphics processor of claim 16, wherein the data cache is configured to invalidate a data entry associated with a position in a buffer when a write is pending for the position in the buffer, producing an invalid data entry.

18. The programmable graphics processor of claim 16, wherein the data cache is configured to read data from the position in the buffer and store the data read in the invalid data entry associated with the position in the buffer.

19. The programmable graphics processor of claim 16, wherein the data cache is configured to update the entry in the data cache when the write to the position in the buffer is completed.

20. The programmable graphics processor of claim 15, wherein the conflict detection unit includes a hash unit.

21. A system for processing fragments, comprising:

means for determining whether a position conflict exists for a fragment, prior to processing the fragment; and

means for deferring processing of a fragment for which a position conflict does exist until the position conflict does not exist.

22. The system of claim 21, wherein the processing includes shading.

23. The system of claim 21, further comprising:

means for permitting a fragment for which a position conflict does not exist to bypass a fragment for which a position does exist.

24. The system of claim 21, further comprising:

means for storing at least a portion of a position associated with a fragment.

25. The system of claim 21, further comprising:

means for storing data corresponding to a region including a position associated with a fragment.